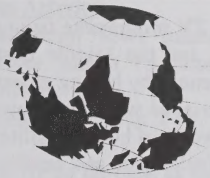
SPACE
Pacific



Membership Notes:
Vol. 4/No. 3
December 21, 1998

Membership Teck Notes is a confidential communication to all members of SPACE. The contents are, where applicable, Copyright 1998 by SPACE Pacific Ltd. Contacts: SPACE Pacific at PO Box 30, Mangonui, Far North, New Zealand (tel 64-9-406-0651 / fax 64-9-406-1083).

-INSIDE THIS ISSUE-

pg. 1 ▪ SPRSCS '99 Announcement; pg. 3 ▪ After SatFACTS for December Close - New Signals; pg. 4 ▪ Mark Long - Encryption Basics; pg. 10 ▪ Cryptography - The Secret Language of Spies; pg. 14 ▪ Using PIC Microcontrollers (in Amateur Radio Projects); pg. 23 ▪ The Dr. Dish SPRSCS '99 Fund Raising Project; pg. 25 ▪ SPRSCS '99 Registration Packet

The South Pacific Region Satellite & Cable Show is being held at the location of Far North Cable TV Ltd., as it was last year (Mangonui, at the top of North Island, New Zealand). The primary goal of this year's gathering is the creation of television programming for distribution to members and others through two and possibly three satellites during 1999.

SPACE Pacific Report (SPR) is a one hour television programme (on occasions two hours) to be distributed to a schedule using the facility of Filipino broadcaster KIBC (AsiaSat 2, 3940/1210 Vt with Msym 26.655 and FEC of 2/3). The first broadcast is tentatively scheduled for March. Negotiations with Nauru broadcaster Sports Pacific Network (SPN) could also result in distribution of the same programming on Intelsat 180E as well by April.

SPACE Pacific has become custodian for more than 300 hours of industry training and event coverage tape spanning the period from 1975 onward. This "archival" tape has immense historical value as it documents the development of "home TVRO" from the very first C-band dish systems (in the UK and North America) through the implementation of small Ku-band dish systems. The actual pioneers (Steven Birkill, Taylor Howard, Bob Coleman to name three) appear "on tape" and describe what it was they did to first access geostationary satellites. Additionally, there are dozens of hours created "on location" at such spots as Sri Lanka (where satellite Godfather Arthur C. Clarke was visited), RCA (GE) uplink sites where management and engineers give an extensive tour of the flight control centre for geostationary satellites, and many-many more.

To make history more interesting, we are preparing current event pieces as well, to illustrate the changes that have taken place over the years as well as the current state of the various technologies. The purpose of the TV programme is to communicate, as accurately as possible, the technology and the events of our industry. SPRSCS '99 will be a four day "taping session" during which various industry expertise will be laid down on videotape and these tape segments will be utilised within SPR during 1999 and probably into 2000. A similar (monthly) TV programme created in Europe by Senior Technical Editor Christian Mass of Germany's **Tele-Satellit Magazine** is a sheduled participant in the SPRSCS '99 taping. Mass, also known as "Dr Dish" in the field, is scheduled to fly to New

SPACE Membership Teck Notes is compiled by SPACE Pacific to assist members in their own understanding of the ever changing technology in the fields of satellite communications and reception. Some material appearing here has been previously published outside of the Pacific + Asian regions and is made available to members for purposes of building a reference library. SPACE offers members the opportunity to advance their own formal training through the Mark Long SPACE Pacific "study at home" technical courses, each of which makes possible a certificate upon successful completion of a course of study. Additionally, during the March 24-27 SPRSCS '99 Membership Conference, members may take the course "live" from instructor Mark Long (see pages 25 and 26 here).

Zealand and be a part of SPRSCS '99 and the TV programme taping. His monthly Dr Dish TV Show distributed throughout Europe, North Africa and the Middle East, has already used segments of material from the SPR "library" and we will benefit by having an exchange programme allowing SPR and SPACE to utilise some of his taped material as well in our own show. The end result of all of this is simply that for the first time the entire world of home satellite TV installation becomes a single entity through the magic of our creative use of television and satellite distribution.

Costs

This is an ambitious project. KIBC (and possibly SPN later in the year) have agreed to provide air time at a very reasonable cost. But to put together a one hour long "report" will require tens of hours of semi-professional editing and video production time. All of this has to be paid for by SPACE. Suppliers to the industry have agreed to "sponsor" some of the "below the line" production costs and of course we are donating our own expertise and time to make this a reality. This is not a profit centre!

All revenues from SPRSCS '99, less only the absolute costs of the venue and Mark Long expenses, will go directly into the SPACE Pacific Report (SPR) fund for 1999. In recognition of those who attend SPRSCS '99 and thereby provide funding assistance for SPR, end of TV programme "credits" will acknowledge all supporters to the "SPR Fund"; a form of "television thank you card" that will record for posterity that certain people were behind the project with their cheque books or credit cards.

To fly Christian Mass out to New Zealand, we are creating a separate "Dr Dish" fund and are asking members to donate funds to this expensive undertaking. This can be done with a credit card and we ask that the amount donated be in multiples of $25 in New Zealand or Australian or US currency to allow us to keep it properly recorded. What this gets you - whether you attend SPRSCS '99 or not - is a permanent mention as a "Dr Dish Sponsor" in both the SPR TV show *and* the Dr Dish TV show in Europe. It will also get you a special "Membership Discount" on various videotape products to be offered to members effective with our next Teck Notes Newsletter. *Please see form at rear of this issue for donation to the Dr Dish fund.*

Why Attend SPRSCS '99?

Why would you be inclined to spend money to attend a trade show that is going to be a "4 day TV shoot?" Fair question. First of all, you will "mingle" with several dozen people just like you from throughout the Pacific, Asia and Europe. People who make a living in (or live for) satellite TV. People like Av-Comm's Garry Cratt who pioneered equipment for home TVRO installations in the Pacific, Robin Colquhoun who has forgotten more about IRDs than most of us will ever know and that super teacher Mark Long. Each person attending will be given three options:

#1) Just be an observer and "hang out" for the TV shoots

#2) Participate as a volunteer production person helping with the audio and video production of the TV tapings

#3) Participate as an in-front-of camera person during group discussions designed to bring out field experience (topics such as "Maximising Cross Pole Attenuation" and "Trouble Shooting Nokia IRDs" will be scheduled programme shoot topics).

If you elect to just observe, your financial support (the fee you pay to attend) will be acknowledged in TV programme credits. If you get behind of or in front of a camera, you will get double credits for your assistance. Five, ten years down the road - these TV programmes we are creating during SPRSCS '99 will become an important part of your own "historical perspective" on the development of satellite TV in the Pacific. And you will be proud to have been a part of their creation as the 1999 SPRSCS '99 programme effort becomes a living chapter of "how we were" at a particular point in time.

Plus - Mark Long's SPACE Certificate Courses

As is standard practice, world famed teacher Mark Long will be on hand during SPRSCS '99 to conduct his twin two-way cram courses leading to certification by SPACE. See registration details on page 25 here.

Post December SatFACTS Satellite Activity...

Although we attempt to stay away from "news" material in Teck Notes, there has been an unusual amount of new digital activity since SatFACTS closed for the printer December 7. The current reports (to December 16) are as follows:

ApStar 2R: Hallmark + Kermit have been FTA (SA PowerVu format) since December 10 (3720/1430Hz-19.510-3/4).

Austar (Service): As predicted in SF as far back as October, Austar is adding three movie channels from Optus to line-up offering Movie One, Movie Extra and Movie Greats from 1 January. They will be free of charge to existing Austar subscribers through February 28th and then available as an optional movie package at $10.95 per month. Also being added - Plus Weather 21, an Australian weather service, which becomes a new channel in the Austar package (no added charge for this one).

JcSat-3: Sudden life from 128E. Try 3990/1160 Vt for 6.109 - 1/2 and "NIC-J [1/2] and GAOR" - whatever that is!

Indovision: The Jakarta based company is telling its customers they have "serious business disagreements with the service management being handled by Star TV Asia." From Hong Kong, we are told that Star TV expects to "take over" the Indovision service by moving its estimated 30,000 users from C2 to AsiaSat (which AsiaSat is not mentioned - for a reason) and then when AsiaSat 3S is operational at 105.5E replacing AsiaSat 1, Indovision subscribers will become clients of the new Star TV Asia digital service through 3S. On December 15th, all but a few of the C2 Indovision channels were shut down while simultaneously S-band service on Cakrawarta expanded to 30 programme channels. We checked with Jakarta and learned there is a sudden rush to move C2 clients to S-band, refitting feeds and LNBs, repointing dishes. This is going on as you read these words. That C2 service was cut off *before* the S-band conversions began suggests strongly Indovision is fighting with Star TV Asia for control of the Indovision customers and the move to S-band was forced upon them by Star's decision to appropriate Indovision customers for Star TV.

Palapa C2: 4000/1150 Hz (5.630-3/4) TVNZ (NZ) Bangkok feeds. 4100/1050 Hz (6.498-3/4) McGibben feeds (2). C Sky Net Taiwan has begun on 3750/1400 Hz with as many as 21 "programme channels" loading (of which 10 are FTA TV, 1 is CA, 2 have no data streams and 8 are radio [audio only] service channels, some heavily distorted). Enter 21.095-5/6 and note that (as of our press deadline) P5 is multiple language movies (Sun Movies, commercially sponsored), including English, with Cantonese sub-titles. There is a great deal of Taiwan TV programming in this bouquet - whether it stays FTA or will be accessible for subscribers is unknown.

PAS-8 Ku: Reports, very little other information, of test CA signals on Australian Ku beam at 12.366 and 12.526 Hz with 28.000 (some say 28.800) and 2/3 (some say 3/4). See SatFACTS December, p. 29.

Thaicom 3: ITC (Information TV Channel) began regular transmissions on December 9 using 3569/1581 Hz (10.200 - 3/4).

Sky (NZ) TV: Currently providing digital service from 12 programme channels out of total of 18 using single transponder on Optus B1. Until existing universe of analogue decoders can be replaced with new Pace or Zenith digital IRDs, the following programme channels are not available: #4 - TNT, #5 - UK TV, # 10 - ESPN, # 12 - National Geographic, # 14 - Animal Planet, # 17 - Sky News.

Launch News: Insat 3A (exact location unknown) moved ahead for March launch. One Russian source says AsiaSat 3S launch has been moved forward to February 21 (was scheduled for "mid-March").

SPRSCS '99
March 24 - 27 1999

Be an "observer" or "participant" in the creation of television programming for SPACE Pacific Reports. Attend one or both Mark Long Certiufication Courses. Details pages 25-26.

THE MOVE TOWARD ENCRYPTION

It wasn't until the advent of the "Satellite Age" that broadcasters were forced to make major changes in the way that they transmitted their TV signals. On November 5, 1982, Canadian Satellite Communications Inc. (CANCOM) became the first full-time TV programmer to encrypt four of their TV services relayed by Canada's Anik D1 satellite. By the end of 1985, more than one million satellite TV systems were installed and operating across North America. Several other programmers who were using satellites for cable TV distribution purposes began to consider how they might prevent the growing C-band satellite dish market from accessing their pay-TV signals.

On January 15, 1986, Home Box Office became the first U.S. pay TV operator to use encryption when it began scrambling its HBO and Cinemax feeds to cable TV affiliates nationwide. Other programmers, such as CNN Headline News, Showtime and the Movie Channel, also followed suit. Due to legislation passed by the U.S. Congress in 1984, however, all pay TV programmers using encryption technology were obligated to offer their services to any home dish owner who was willing to pay for authorized access.

THE TECHNOLOGY OF ENCRYPTION

All broadcast TV encryption systems have three primary components: an encoder, an authorization center, and a universe of low-cost, individually addressable decoders.

The encoder, which is located at the satellite uplink facility, converts the analog TV signal into an encrypted message that must be decoded before it can be displayed on a standard TV set. The encoder is also linked to a computerized authorization center that can process subscription orders and control all decoders in the system.

Total control is achieved by assigning a unique address code to each subscriber's integrated receiver/decoder or IRD. The authorization center uses this code to turn on, or turn off, any individual IRD, or even to control a selected group of set-top boxes. The authorization center, for example, can "black out" specific regions of the country when a particular programmer is transmitting a sporting event for which it does not own the national broadcast rights.

Whenever I refer to a set-top box as an analog IRD, I am actually referring to the type of satellite signal that the IRD is capable of receiving. As we shall see below, the so-called analog IRD must be capable of processing a variety of digital instructions before it can deliver any analog-based satellite TV program to the viewer's TV set.

VIDEOCIPHER RS

In North America, the VideoCipher-RS (for Renewable Security) encryption system developed by General Instrument is widely used by the satellite broadcasting community to transmit pay TV subscription services to receiving dishes equipped with analog-style, set-top boxes. The VideoCipher RS encryption system, or "VCRS" as I shall call it hereinafter for brevity, can accommodate up to 256 "tiers" of programming. Each tier can be used to control access to either a single pay-TV service or a group of pay-TV services that are marketed as a unified program package. Moreover, VCRS has the capability of individually addressing up to 50 million set-top boxes. VCRS also can employ electronic countermeasures, whenever required, to disconnect any set-top boxes that have been illegally

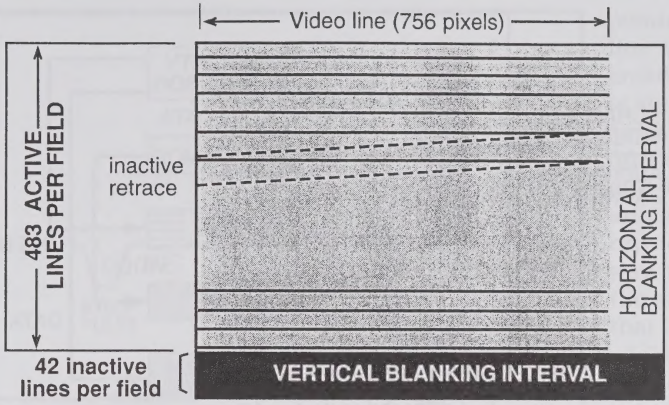


Fig. 7-4. 525-line NTSC horizontal and vertical blanking intervals.

modified to receive encrypted TV programs for free.

VCRS also features an electronic "smart card" that provides the essential conditional access (CA) data that the IRD needs before it can decode the encrypted signals. This smart card, which plugs into a CA card reader slot located on the front panel of the IRD, contains a single Large Scale Integrated Circuit (LSIC) chip that holds the mathematical algorithms or "keys" that unlock each IRD in the system.

Encrypting the Video Signal. All TV sets, whether NTSC, PAL or SECAM, contain a cathode ray tube (CRT) that uses an electron gun to shoot a stream of electrons at the interior wall of the TV screen. This beam of energy excites the screen's phosphor coating, which produces the images that we see.

When the electron gun reaches the end of each video line, it must be told when to switch off so that it can invisibly sweep back across the screen and begin to trace the next video line. A sync pulse located in the TV signal's "horizontal blanking interval" (the horizontal timebase) provides this information. Another sync pulse located within a corresponding "vertical blanking interval" (the vertical timebase) instructs the electron gun to switch off and return to the top of the screen.

The VCRS encoder removes both of these sync pulses, with the result that the TV set has no way of determining precisely when to end one video line or field and begin the next. The encoder also inverts the TV signal's luminance component to a negative state and also shifts the location of the chrominance signal component, further confusing the TV set.

To make amends, the encoder digitally delivers a set of instructions that tells the IRD exactly when to insert the missing sync pulses, where to find the missing color information, and just how to return the luminance component of the video to the normal "upright" state. However, this digital set of instructions is encrypted so that the signal can only be resolved by those set-top boxes that have the electronic keys required to correctly read these instructions.

The General Instrument factory electronically programs a unique twelve-digit unit address into each IRD that it manufactures. The TV screen will display each IRD's address code whenever the viewer uses the remote control to enter the "SETUP 1" command. Each programmer requires this address number whenever a new subscriber wants to order an encrypted TV service.

The programmer provides the IRD's address code to the General Instruments DBS Authorization Center in La Jolla, California. The Authorization Center in turn will send an authorization message over the satellite that provides

the IRD with the set of instructions that it needs to begin decoding the service. Alternatively, the Authorization Center can remove the authorization message from the data stream to shut off any individual IRD. Either process can be completed within a matter of seconds.

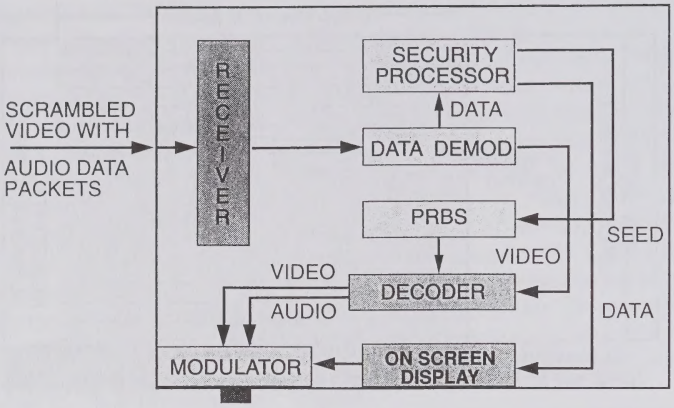


Fig. 7-5. Block diagram of an intergrated receiver/ decoder (IRD).

VCRS also includes a "SETUP 0" command that will display diagnostic data on the TV screen. In the event that subscribers are unable to receive a TV program service for which they have already been authorized, the Authorization Center may ask the subscriber to provide some of these codes to assist in determining the cause of the service interruption.

Encrypting the TV Audio. The VCRS encoder is not merely satisfied with transforming the video component of the TV signal into a jumbled mess. It also digitizes the audio and then inserts the resulting digital bit stream into the encrypted TV signal's horizontal blanking interval (HBI), which has enough space to accommodate one stereo audio channel plus a utility data channel.

The encoder also encrypts the bit stream by adding each digitized audio sample to a sequence of binary numbers that have been generated by the encoder's pseudo-random binary sequence generator, or PRBS. This random binary sequence is the "electronic key" that the IRD needs to restore the audio to its original state. In the case of all U.S. encryption systems, this is the 56-bit DES (for Digital Encryption Standard) algorithm mandated by the U.S. National Bureau of Standards.

To generate this 56-bit key, the IRD must also contain a PRBS that can be synchronized with the PRBS at the encoder so that both units are producing identical strings of random numbers. The encoder transmits information over the satellite, called the "seed," that provides each IRD's PRBS with the required synchronization information.

The encoder also encrypts the seed so that the IRD must also have an electronic key for unlocking this synchronization information. The smart card is the system component that supplies this function.

Similar to the digital set-top boxes discussed in the previous chapter, VCRS also transmits forward error correction (FEC) data to assist the IRD in detecting and correcting errors caused by link noise. The FEC data is transmitted within the HBI along with the audio's digital bit stream. VCRS-compatible, set-top boxes can detect and correct single-bit errors as well as conceal or "mask" double-bit errors.

OTHER ANALOG ENCRYPTION SYSTEMS

There are several other encryption systems that satellite dish owners in

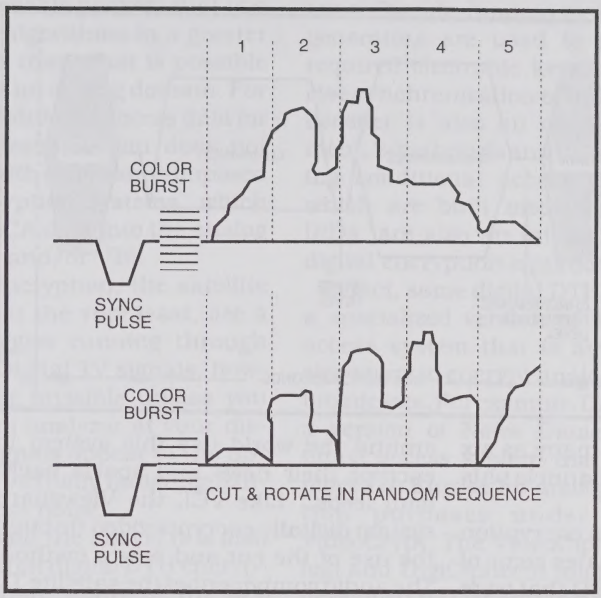


Fig. 7-6. The cut and rotate method of encryption.

the Americas may encounter as they travel along the satellite superhighways. Let's take a brief look at a few of the more commonly encountered ones.

VideoCipher I. The "VCI" system is exclusively used to encrypt satellite TV signals that are not intended for viewing by the general public, such as news and sports "back-hauls" from the site of a live event to a TV network's home studios. There is no such thing as a consumer-grade IRD for VCI.

VCI is an improvement over VCRS in that it digitally encrypts both the video and audio components of any satellite TV broadcast. VCI uses what is known as the "cut and rotate" method for encrypting the video signal. The encoder samples each video line and converts each line segment to an equivalent digital message. The encoder also determines the cut points for each digitized line and then rotates each line segment so that it can be inserted between alternate cut points. The location of the cut points varies from line to line and all vertically oriented picture information is stepped back and forth across the screen in a sequence that changes from one field to the next. The audio component is encrypted much in the same way that VCRS encrypts the audio for the services that it handles.

Like VCRS as well as most other modern encryption systems, VCI uses synchronized pseudo-random binary sequence generators at the uplink and downlink locations to generate the algorithms required for encrypting the video and audio signals. With VCI, the IRD receives the set of instructions needed to perform complementary cut-and-rotate operations that reassemble the image into its original state. The VCI encoder uses the vertical blanking interval (VBI) to send instructions to the IRD concerning the location of the cut points.

B-MAC. Developed by Great Britain's NTL and exclusively licensed to Scientific Atlanta, B-MAC (for Multiplexed Analogue Component, Type B) is used by satellite service providers worldwide to encrypt their TV program services. At one time or another, B-MAC has been used for DTH operations in several countries including Australia, Indonesia, South Africa and the United States. Within North America, however, B-MAC hasn't been used for DTH applications since Primestar switched to an all-digital delivery system in the mid-1990s.

B-MAC uses an encryption technique known as the "line translation" method, where each video line is delayed in a

random fashion by several microseconds. This creates the familiar cross-hatched, diamond-shaped patterns that satellite dish owners the world over encounter occasionally.

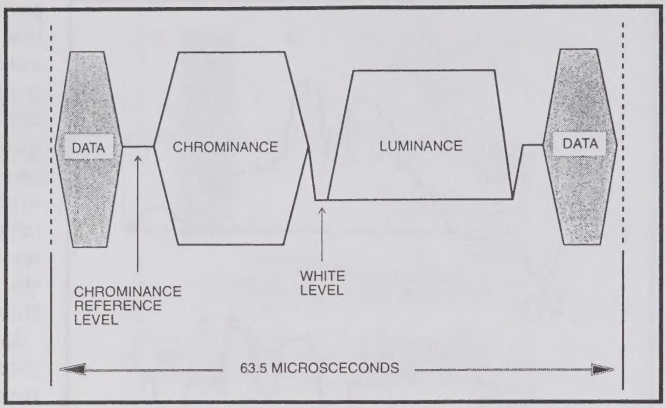


Fig. 7-7. The sequential transmission of the chrominance, luminance and data information is one of the key features of B-MAC.

The B-MAC encoder uses the VBI to send addressable packets to its IRD universe, while the HBI is used to transmit as many as six digitally-encrypted audio channels plus one utility data channel.

In addition to its role as an encryption system, B-MAC also eliminates some of the inherent flaws with NTSC that were mentioned earlier in this chapter. For example, B-MAC totally eliminates the picture artifacts caused by the "cross-talk" between the luminance and chrominance components of the NTSC signal. With B-MAC, the chrominance information is transmitted within one-third of the active scanning time, with the luminance information contained in the remaining two-thirds.

Viewguard. This is yet another encryption system that is not used for DTH applications. Broadcast networks around the world use this system to encrypt their news and sports backhaul feeds. Like VCI, the Viewguard system digitally encrypts video through the use of the cut and rotate method. The audio component of the satellite TV signal also is digitally encrypted.

DIGITAL DTH ENCRYPTION

The available encryption techniques increase dramatically whenever an analog TV signal is transported into the digital domain. As I pointed out in the last chapter, DVB-compliant digital DTH systems can be rendered unavailable by restricting access to compatible set-top boxes as well as any information concerning the symbol and FEC rates in use. Digital DTH signals also are trans-

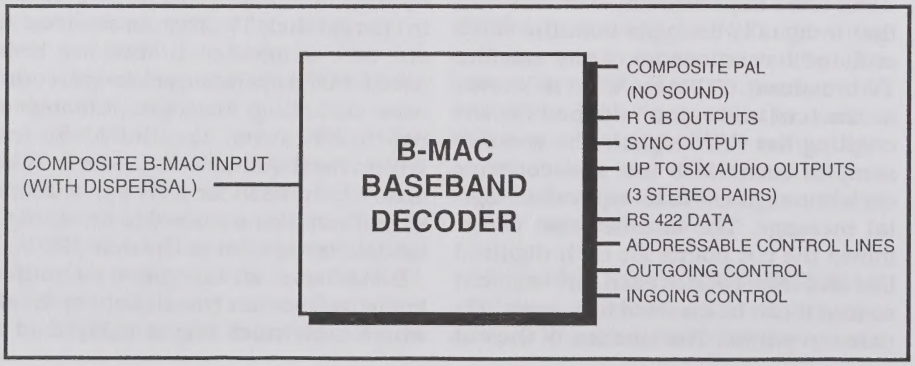


Fig. 7-8. B-MAC baseband decoder interfaces.

mitted in addressable packets that can be subjected to algorithms in a greater number of ways than what is possible when working in an analog domain. For example, the conditional access data for a digital DTH transmission does not have the bandwidth constraints imposed on analog encryption systems, which must insert the CA data into the analog TV signal's HBI and/or VBI.

With analog encryption, the satellite TV viewer will, at the very least, see a jumble of squiggles running through the TV screen. Digital TV signals, however, are all but invisible unless you have a spectrum analyzer at your disposal. Digital signals appear to the uninitiated IRD as virtually indistinguishable from random noise.

Now, having said the above, it is also fair to point out that digital DTH encryption systems share many of the key aspects that have already been described with regard to analog encryption systems. Pseudo-random binary sequence generators are used to generate the required electronic keys, and the precise synchronization of the encoder and decoder is also an essential requirement. Smart cards and their corresponding conditional access (CA) readers, which are built into all digital DTH IRDs, are also an integral part of the digital encryption equation.

In fact, some digital DTH services use a specialized version of a conditional access system that is already in use elsewhere to encrypt analog satellite TV broadcasts. For example, DIRECTV uses a version of News Datacom's conditional access system that also is employed in Europe for analog TV encryption purposes under the name VideoCrypt. The VideoCipher RS (analog) and DigiCipher II (digital) systems developed by General Instruments also share many common elements.

MELBOURNE SATELLITES P/L

ACN 065 270 733
Established 1992

Direct importers and suppliers of the following world renown products.

KTI	ECHOSTAR	GARDINER
ORBITRON	SCIENTIFIC ATLANTA	CALIFORNIA AMPLIFIER
PATRIOT	NOKIA	MTI
JONSA	IKUSI	CHAPARRAL
ANDREW	DX	ADL

Full range of cables, splitters and connectors.

Phone, fax or write for our 1998 Product *SAT*alogue.

Phone: (03) 9738 0888
Fax: (03) 9729 8276

Sales/Warehouse - 84 Bayfield Road, Bayswater Victoria
Construction Services - 13 Elsum Avenue, Bayswater Victoria
Postal Address - PO Box 901, Bayswater 3153 Victoria Australia

www.melbournesatellites.com.au

Cryptography – The Secret Language of Spies

By Dan Veeneman

"23187... 46982... 69335... 15948..."

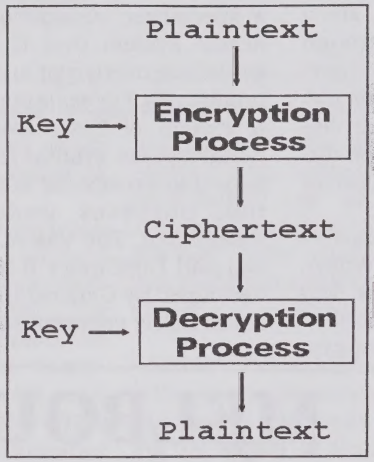
If you've listened to shortwave for any length of time you've probably run across them. A mysterious voice announcing a long series of numbers, perhaps destined for some unknown spy in a foreign land. Although shrouded in secrecy, these numbers are assumed to be some kind of encrypted message, making sense only after being decoded by some clandestine listener.

Cryptography is the science of secure communication in the presence of adversaries. In any encryption system, the message to be sent, called the *plaintext*, is encrypted in some way to produce *ciphertext*, which is then transmitted. Encryption systems typically require a secret piece of information, called a *key*, in order to scramble the message. On the receiving end, the key is used to decrypt the ciphertext and produce the original plaintext. Should the ciphertext be intercepted, an eavesdropper would be unable to decrypt it without the proper key, even if the encryption method was known.

Despite the lack of concrete details about these mysterious numbers stations, over the years it has been assumed that the numbers are really messages encrypted using a *one-time pad*.

One-Time Pad

The only provably secure cryptographic system was created by Gilbert Vernam in 1917, who originally developed it to protect electronically transmitted messages. His invention automatically enciphered individual characters as they were entered into a teletypewriter as well as automatically deciphering them at the receiving end. This kind of character-by-character encryption is called a *stream cipher*, and Vernam's particular method is unbreakable if used correctly. His invention is now referred to as the one-time pad cryptosystem, and has been extended to both manual and computerized operations.



The key in a one-time pad system is a series of random numbers. Vernam's invention used long strips of paper tape, but in the manual version of the system numbers are printed on small pads of paper that are easily concealed. Each sheet of paper in the pad is used only once, then discarded and permanently destroyed. (It has been reported that the Central Intelligence Agency has a special type of paper that turns into chewing gum when it contacts saliva.) There are two copies of the one time pad: the person wishing to send a message and the intended recipient both have identical pads.

An Example

With all that in mind, let's take a look at a contrived example.

Spy headquarters is preparing to send an operative out into the field. During training the operative learned that most cryptographic systems involve mathematical operations, where it's much easier to deal with numbers than letters. Spy HQ has established a simple coding scheme where each letter of the alphabet is assigned a number. The letter A has a value of 1, the letter B has a value of 2, and so on up to Z, which has a value of 26 (see Table 1).

Table 1: Letter-number pairings.

Letter	Value
A	1
B	2
C	3
...	...
Z	26

In addition to a false passport and some local currency, the operative is issued a small pad of paper with a series of random letters printed on each page. This one-time pad contains the keys for the operative to decrypt messages that will be sent in the future. Spy HQ keeps an identical copy of the pad. Some time passes and the operative becomes active in the target country, occasionally listening to shortwave radio for any messages that may be destined for him.

At some point Spy HQ wishes to send a message instructing him to purchase a particular magazine. Using the letter-to-number conversion table, Spy HQ converts the phrase MONITORING TIMES into the numbers 13 15 14 09 20 15 18 09 14 07 20 09 13 05 19 (skipping the space). This is the plaintext message.

To encrypt the message Spy HQ will use a page from the one-time pad. For this example we'll say the next page of the pad has the key SECRET LISTENERS written on it, although

n practice the key would be a random string of numbers. Again ignoring spaces, SECRET LISTENERS is encoded as 19 05 03 18 05 20 12 09 19 20 05 14 05 18 19.

The simplest encryption method is to add the plaintext number to the corresponding key number for each letter in the message (see Figure 1).

```
13 15 14 09 20 15 18 09 14 07 20 09 13 05  19
MONITORING TIMES
19 05 03 18 05 20 12 09 19 20 05 14 05 18  19
SECRET LISTENERS
-- -- -- -- -- -- -- -- -- -- -- -- -- -- --
32 20 17 27 25 35 30 18 33 27 25 23 18 23  38
```

FIGURE 1: *Adding each plaintext number to the corresponding key number.*

Since the letter encoding scheme only goes up to 26, Spy HQ "wraps around" the sums that exceed 26 by subtracting 26 and using the difference (see Figure 2). This operation is called taking the modulus and is done in many situations involving counting. (For instance, there are twelve hours on the face of a normal clock, and after reaching 12 we "wrap around" and continue counting at 1. This is called taking the hour "modulo 12.")

```
32 20 17 27 25 35 30 18 33 27 25 23 18 23  38
-26 -26 -26-26 -26-26 __ __ __ __ __ -26
06 20 17 01 25 09 04 18 07 01 25 23 18 23  12
```

FIGURE 2: *Taking the addition result modulo 26.*

So Spy HQ's encoded message is 06 20 17 01 25 09 04 18 07 01 25 23 18 23 12 (the nonsensical FTQAYIDRGAYWRWL), which is transmitted over the air at a time when the operative is scheduled to be listening.

On the receiving end the operative uses the identical key from his one-time pad to recover the message. Each key value is subtracted from the corresponding ciphertext letter to produce the plaintext (see Figure 3).

```
06 20 17 01 25 09 04 18 07 01 25 23 18 23  12
19 05 03 18 05 20 12 09 19 20 05 14 05 18  19
-13 15 14-17 20-11-08 09-12-19 20 09 13 05 -07
```

FIGURE 3: *Decoding the received message by subtraction.*

The values that fall below 1 are "wrapped around" to a positive value by adding 26 (Figure 4).

```
-13 15 14-17 20-11-08 09-12-19 20 09 13 05 -07
+26 +26 +26+26 +26+26 __ ____ +26
13 15 14 09 20 15 18 09 14 07 20 09 13 05 12
```

FIGURE 4: *Taking the difference modulo 26.*

This gives the operative the original plaintext message of MONITORING TIMES.

An eavesdropper listening to the same shortwave station would have just the numbers 06 20 17 01 25 09 04 18 07 01 25 23 18 23 12, spelling FTQAYIDRGAYWRWL. If Spy HQ and the operative are following all the rules to maintain security, the best the eavesdropper can do is try out every possible key. This method is called *brute force* decryption, but in a one-time pad system where the key is as long as the message, even that drastic step won't help. The eavesdropper, in fact, will end up with all possible solutions.

For example, decrypting FTQAYIDR GAYWRWL using the key NSNVVHL YTLKCJKM produces the result RACE CARS MONTHLY and the key RSWRJUCFMZECFRT will produce NATIONAL TATTLER, both of which make sense but neither of which are correct. Since any key in a secure one-time pad is equally likely, the eavesdropper has no hope of determining the true plaintext using brute force.

Keeping it Secure

In order for a one-time pad system to remain secure, several rules must be followed:

Example One-Time Pad

```
48173 19839 90183
51834 00182 47865
01983 47362 3    2
60120 98754 2    874
```

First, the contents of the one-time pads must be kept secret. This is fairly obvious, but important. Anyone discovering the pads, if they didn't shoot the owner on sight for being a spy, could make a copy of the pad and decrypt the messages themselves. Intelligence and law enforcement agencies routinely perform "black bag jobs" to covertly break in and steal keys used by embassies, corporations, and even private individuals. This is probably the most damaging situation for the one-time pad user, since the only thing worse than no security is a false sense of security.

Second, the pads must not be reused. Two-time pads are not mathematically secure, and in fact the reuse of pads allowed the National Security Agency (NSA) to break thousands of Soviet KGB and GRU messages in the 1940's.

Third, the contents of the pads must be unpredictable. Regardless of whether the spy keeps the pads a secret and doesn't reuse them, if the numbers are somehow predictable it would be the equivalent of the adversary having their own copy of the pad. Numerous modern software programs that offer "one-time pads" have this weakness — their method of creating "random" numbers is often pre-

NEW RECEIVER
UNIVERSAL SC-50
SUBCARRIER—FM² AUDIO RECEIVER
UNIVERSAL
S-20 C-SPAN M49
SC-50
RECEIVE ALL FM² AND AUDIO
SUBCARRIERS—100 kHz to 9 MHz
Full featured audio services, music, all sports, talk shows, news, religious programming, major radio stations, variety, public radio plus many other services, no fees. The SC-50 audio subcarrier receiver will work with all home satellite systems, 3-minute hookup, simple and quick to tune, 16-character display, 50-channel memory bank, direct frequency readout, covers all FM² and audio subcarrier channels, hundreds of free programming channels.
FOR INTRODUCTORY PRICE CALL: 1 - 614 - 866-4605
UNIVERSAL ELECTRONICS, INC.
Communications Specialists
4555 GROVES RD., SUITE 12, COLUMBUS, OH 43232
(614) 866-4605 FAX (614) 866-1201

dictable, allowing a snoop to reproduce the key and decode messages.

Randomness

Do these numbers seem random and unpredictable: 212, 198, 216, 32, 175, 100? If you think so, you don't commute on Interstate 95 north of Washington, D.C. These are the Maryland route numbers, in order, of the exits off northbound I-95 after leaving the D.C. beltway. They may look random, but they are very predictable once you know the pattern.

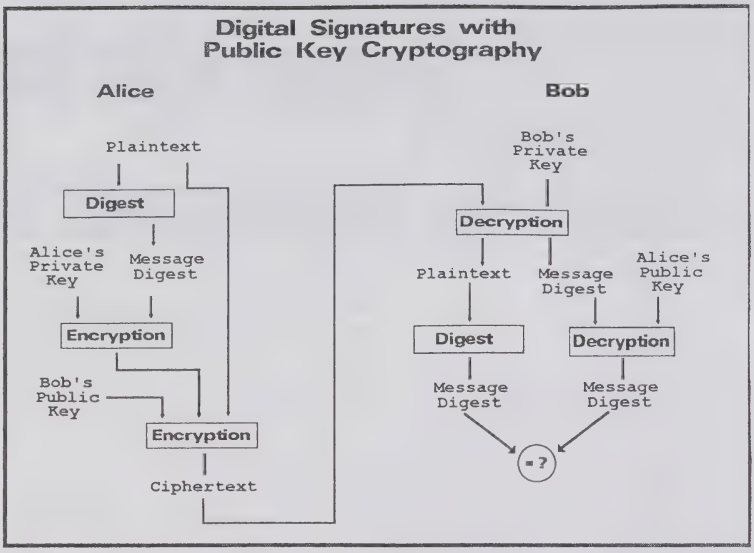
It's surprisingly difficult to generate really unpredictable numbers. For those of you who may be planning to use the random number function available in most programming languages for cryptography, please think again. Just like the Maryland route numbers, the output from these functions look random but are predictable and repeatable. In fact, a couple of years ago a serious weakness in Netscape's web browser was revealed after researchers discovered the random number generator used in the browser was very weak. This weakness would have allowed eavesdroppers to easily decode protected web connections, including the details of any financial transactions.

Key Length

In a secure one-time pad system the key must be as long as the message. If you want to send a message containing two thousand characters, you need a key that is also 2,000 characters long. If you want to encrypt a computer file that is two megabytes (2 million characters) with a one-time pad, you will need a key that is also two megabytes. If you want to protect a digital video signal that carries hundreds of thousands of characters each second, you probably won't use a one-time pad. Creating, delivering, and securing large keys is a very difficult problem, limiting true one-time pad systems to very specific uses, such as clandestine operatives in foreign countries.

For the rest of us, practical limits on the amount of key information we can handle necessitate the use of other encryption systems. In *block ciphers* the key length is much less than the length of the message, and is used in a different way. The plaintext message is broken up into small pieces called blocks, and each block is encrypted by the key. There are a wide variety of block ciphers that operate in different ways, but as a general rule the smaller the key the less secure any message encrypted with that key will be.

Keys in modern cryptographic systems are measured by the number of bits they contain. A bit is the smallest unit of information, either a 0 or a 1. The Data Encryption Standard (DES), a popular cipher, uses keys that are 56 bits long, allowing approximately 72 quadrillion (72,057,594,037,927,936) possible keys. As we'll see, this isn't nearly enough. Skipjack, a cipher invented inside the NSA and classified secret until very recently, uses 80 bit keys. The International Data Encryption Algorithm (IDEA) uses keys that are 128 bits long. Other ciphers of varying strength and complexity exist as well, each with their own key lengths. All other things being equal, one additional bit doubles the amount of work necessary to brute force a solution.



The United States government considers cryptography to be an implement of war, and closely controls the export of strong cryptography. Until recently, ciphers using more than 40 bits of key were usually denied export permits with the justification that such systems were too difficult for the government to crack and would therefore jeopardize national security. Even 56-bit DES has been denied export in many circumstances, despite its availability overseas. In addition, domestic law enforcement representatives have been wringing their hands in public, fretting that the widespread use of strong cryptography would render their wiretaps useless and frustrate their investigative efforts. In an effort to convince law makers of the dangers of strong cryptography, the Federal Bureau of Investigation (FBI) has testified before Congress numerous times about the impossibility of cracking encrypted messages protected by cipher systems such as DES.

DES Cracking

The Data Encryption Standard (DES) is a Federal Information Processing Standard (FIPS) approved by the National Bureau of Standards (now the National Institute of Standards and Technology, NIST) in 1977. In the intervening decades it has been scrutinized by numerous experts in cryptography, none of whom have found a significant weakness in its fundamental design. What is weak is the inadequate size of the key.

As mentioned above, brute force cracking is the process of decrypting a message using every possible key to find the one that works. To motivate practical research, in 1997 a company named RSA Data Security offered a prize for cracking a message encrypted using DES. Hundreds of volunteers ran customized key search software in their spare time on available computers, exchanging results and passing key information over the Internet. The message was broken after five months of effort across several thousand computers. At the beginning of this year the prize was offered again with a different message, and an improved search method yielded the key in only 39 days, again broken by a group effort of thousands of computers coordinated over the Internet.

A third challenge was broken in July by the Electronic Frontier Foundation (EFF), who used a single custom-built machine to crack the code in less than three days. For well under $250,000 the EFF had built a "DES Cracker" from a personal computer and an array of

custom microchips that could break a DES-encrypted message in reasonable time without having to coordinate with anyone. They also showed that the government had been playing fast and loose with the truth in their testimony of how difficult it is to break DES messages.

Public Key Systems

Existing encryption systems fall into two categories. Private key systems, sometimes called secret key systems, use the same key on the sending and the receiving ends. One-time pad and DES cipher methods are private key systems.

Public key systems, on the other hand, make use of a public/private key pair. The public key is published and available for anyone to see while the private key is kept secret by the owner. The public and private keys are mathematically related to each other, but for all practical purposes it is impossible for an adversary given the public key to determine the private key.

For an overview of the use of a public key system let me introduce some characters that have become standard in the cryptologic literature. The protagonists in our story are Alice and Bob, who want to communicate securely while an adversary, say Louis, wishes to intercept and read everything they send to each other.

To send a message in a public key system, Alice creates a message destined for Bob. She encrypts the message using Bob's public key and sends it to him. When Bob receives the message he uses his private key and decodes the message. Since no one else knows Bob's private key, no one else will be able to decode the message.

Although Alice is sure that Bob is the only one that can read the message, how can Bob be sure the message really came from Alice? Perhaps our snoop Louis has effected a *man-in-the-middle* attack and intercepted Alice's message, altering it or replacing it with another message of his own creation before sending it on to Bob.

Digital Signatures

So far we've been talking about *confidentiality*, that is, keeping the contents of the message secret. Cryptography can also help us with *authentication*, that is, proving that a message really came from a particular person.

To foil Louis, Alice can take some additional steps that will prove to Bob she is really the author of the message he receives. Before sending the message, Alice generates what's called a *message digest*, a kind of digital fingerprint that identifies the contents of the message. This digest is based on the exact contents of the message, and no two messages will have the same digest. Alice then encrypts the digest with her private key and sends it along with the encrypted message to Bob. This is the equivalent of Alice signing the letter, and in fact this called a *digital signature*.

As before, Bob receives the encrypted message and decrypts it using his private key. He also decrypts the encrypted digest using Alice's public key. Bob then runs the plaintext message through a message digest function to produce a local fingerprint. If the digest decrypted with Alice's public key matches the digest he just computed, Bob can be sure that the message was not altered and was, in fact, sent by Alice and not an impostor.

All of these features are available in a program called *Pretty Good Privacy*, which was one of the first publicly-available programs to provide strong encryption. PGP uses public key cryptography to protect message confidentiality and assure authentication of the sender. (*MT's Computers & Radio also reviewed a data encryption program called Cyberlock in the August issue - ed.*)

A Challenge

As a final challenge, see if you can decrypt the following one-time pad message. Spy HQ has sent you the message **BABPFQH WZGIRPFNWBCOS** and the next key on your one-time pad is **LISTENING AND LEARNING**.

What is Spy HQ trying to remind you? The answer will appear in my *PCS Front Line* column in the December issue of *Monitoring Times*. For those of you who can't wait, the answer will also be available on my website, which will also have a computer program to assist you. Details below.

Further Reading

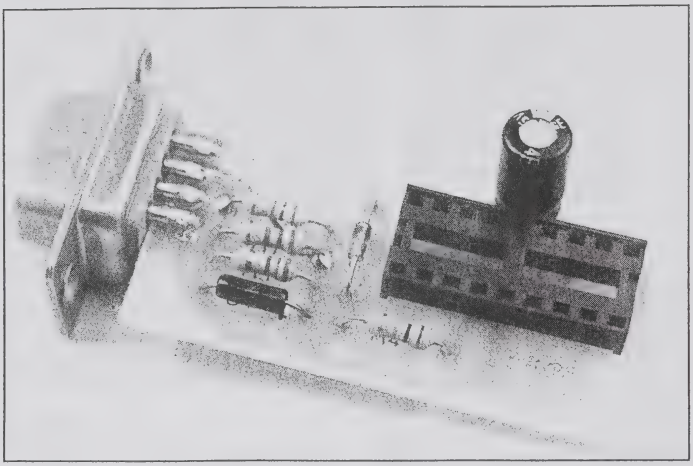
For more on the history of code making and code breaking, David Kahn's *The Codebreakers* is an authoritative account of the use and misuse of cryptography.

For those more technically inclined, Bruce Schneier's *Applied Cryptography* is one of the most popular books detailing modern cryptographic systems and how to implement them.

The American Cryptogram Association is a non-profit, volunteer organization dedicated to the creation and solution of cryptographic challenges of various sorts. More information on the ACA is available by writing to ACA Treasurer, 1118 Via Palo Alto, Aptos, CA, 95003.

All of this information and more is available on my website at **www.decode.com**.

Using PIC Microcontrollers in Amateur Radio Projects



These simple projects should whet your appetite to learn more about the little PIC microcontrollers you see so frequently.

Computers are now embedded in most consumer electronics products. Today, it's virtually impossible to buy a commercially made Amateur Radio transceiver that doesn't include at least *one* microprocessor. Recently, microprocessors have even begun to find their way into Amateur Radio home construction projects. In 1997, for example, nearly half the issues of *QST* included construction projects designed around several different computers on a chip![1-7] [*And we've got more in '98, too!*—Ed.][8-10]

The reason for this is simple: When using embedded microprocessors, it's possible to make equipment much more capable and much cheaper. And—contrary to the initial impression of many hams—embedded microprocessors make project design *easier*, not more difficult.

This article presents an introduction to using one of the simplest and cheapest microprocessors, Microchip Technology's Peripheral Interface Controllers—PICs—in Amateur Radio applications. I'll provide you with the necessary background to begin using PIC chips and give you some pointers on where to learn more about them. Most importantly, it will describe how to build a circuit that allows *you* to program PIC chips yourself — at a cost of less than $5! The next time you want to build a *QST* project that includes a PIC chip, if the source code is available, you'll be able to *program your own chip* instead of paying someone to do it for you!

What Embedded Microcontrollers Do

An embedded microcontroller can be thought of as a tiny computer that receives data, makes calculations or decisions based on that data and then acts in response. It interacts with the rest of the world through pins that can be configured as inputs *and* outputs. Configuring a pin as an *input* means that the microcontroller can *read* the pin to determine whether the voltage on it is high or low. When a pin is *high*, it means that 5 V is applied to it. When the pin is *low*, it means the pin is at ground potential. When a pin is configured as an *output*, it means that the microcontroller itself can make the pin high (+5 V) or low (0 V).

The microcontroller acts on a set of instructions (a *program*) that determine how the microcontroller converts these input signals into output signals. It is incredible the range of functions that these microcontrollers can perform! That's especially true when you consider that the chip's behavior is limited solely to finding out whether input pins are high or low, then setting output pins either high or low in response!

Inputs to the microcontroller might be pushbuttons (or arrays of buttons in a keypad), sensors of various types such as temperature, pressure, or acceleration (fed through an analog-to-digital converter [ADC]), or a serial or parallel data stream from any device capable of generating serial or parallel data (DTMF decoders, radio computer ports, PCs, etc). As long as the information can be presented to the microcontroller as a high or low signal on one or more input pins, the controller can recognize the information and perform predefined functions in response.

By using transistor switches or relays, the microcontroller outputs can switch external devices on and off, generate sounds, or send serial or parallel data to control other devices. Text or data from the microcontroller can be displayed on an LCD panel, or sent to a speech synthesis chip to be read aloud. Recent microcontroller projects in *QST* include a repeater controller (see Note 1), a CW IDer (see

Note 8) and a remote base controller (see Note 10). Each of these projects would have been *vastly* more complex, less capable and much more expensive if embedded microcontrollers had not been available. And think of it: We have only begun to scratch the surface of things that can be done with these ICs!

The Microchip PIC

Microchip Technology's series of PIC microcontrollers are among the most widely used by experimenters.[11] They are quite cheap and relatively easy to use. Microchip makes an assortment of these processors that hold differing program sizes and amounts of data. Some of these ICs include such features as on-board ADCs, serial ports, large numbers of input or output pins and multiple timers. To program these chips, you need a special programmer. To erase *most* of them, you need an ultraviolet eraser. In addition to the cost of the eraser (about $50), erasing one of these PICs takes about four minutes. So, if you are debugging a program by running it, checking how it functions and then making changes, getting to the finished project can be painfully slow.

Fortunately, Microchip has developed one series of chips that is *electrically* programmable and erasable. These chips can be programmed and erased over and over again—electrically. Not only does this make the process of programming and reprogramming easier, it also means the time and expense of using a UV eraser can be avoided.

Until recently, the most common version of these electrically programmable and erasable products was Microchip's 16C84. It has been largely superseded by their 16F84. In the 'F84, Microchip replaced the 16C84's EEPROM with flash memory. Operationally, both ICs are quite similar.[12] The 16F84 is nearly pin and code compatible with the 'C84, and has more room for data. It is also cheaper. In single quantities, the 16F84 costs about $6. If you buy 25 of them at a time, you can cut that cost by a third. Because of its ease of use, the 16F84 is an ideal chip to use when learning about PIC microcontrollers.

A Road Map to the 16F84

Let's dive right in with some projects—I'll explain things as I go along. Figure 1 shows a very simple circuit for experimenting with the 16F84 PIC: an LED and a piezo speaker. The PIC's pins labeled RB0-RB7 and RA0-RA4 are general input/output pins that the IC's program can specify as receiving or sending signals. In this simple circuit, we use only two outputs: RB0 and RB1. Through RB0, we can make the PIC light an LED. RB1 has been hooked to a small piezo speaker to allow experimentation with sound. The piezo speaker is an excellent choice for this application because it can be hooked directly to the output pin of the PIC to provide a fairly loud sound.

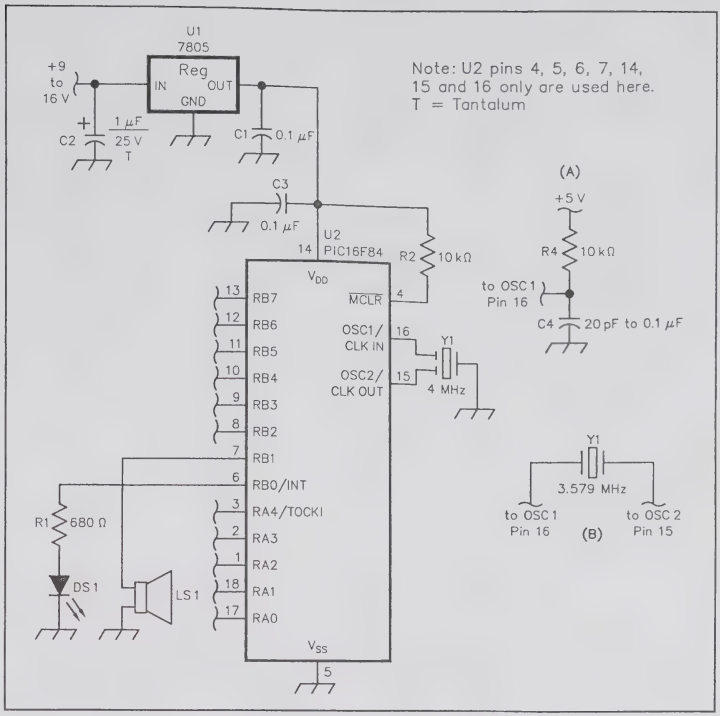


Figure 1—Schematic of the PIC sound-experimentation circuit. Unless otherwise specified, resistors are 1/4 W, 5% tolerance carbon-composition or film units; equivalent parts can be substituted. RS part numbers in parentheses are Radio Shack; DK numbers are Digi-Key.

C2—1 µF, 25 V tantalum (DK P2059; RS 272-1434)
LS1—Piezo speaker element (RS 273-091)
U1—7805 12 V, 1 A positive regulator (DK NJM7805FA; RS 276-1770)
U2—PIC16F84 microcontroller (see PIC Resources sidebar)
Y1—4 MHz resonator (DK PX400) or 3.579 MHz crystal (DK CTX049, HC-49 holder)

The rest of the parts in this circuit are needed in every PIC circuit to make it work. Positive 5 V is always applied to the V_{DD} pin and ground to the V_{SS} pin. Figure 1 shows how to use a 7805 voltage regulator to convert a 9 to 16 V supply to 5 V. An easy alternative is to buy a battery holder that holds four AA cells and use them to power the circuit. Although four AA cells deliver 6 V rather than 5 V, the PIC will handle that voltage level without difficulty.

The PIC needs a clock source connected to pins **OSC1** and **OSC2**. We have a number of clock options. The simplest is an RC timing circuit (see insert A of Figure 1), but its clock rate is slow and it does not provide a high-stability timing source. If you simply want to light an LED, the RC circuit is fine, but realize that it is an inappropriate configuration for tasks that require more accurate timing (such as serial data communication). You can also use an external oscillator, but this is a more expensive solution than is really necessary. A crystal can be connected between the **OSC1** and **OSC2** (see insert B of Figure 1). Microchip recommends including small-value capacitors on both crystal leads, but most users have found this to be unnecessary. TV colorburst crystals (3.5795 MHz) are readily available, inexpensive and work well in this application.

The cheapest way to provide a stable clock, however, is to use a ceramic resonator (shown connected to U2 in Figure 1). These resonators are extremely small parts (smaller than crystals) and generally cost less than $1 each. (Make sure you get resonators with integrated capacitors.) To use a resonator, simply hook its outside pins to the PIC's **OSC1** and **OSC2** pins and the resonator's middle pin to ground.

Finally, you need to apply 5 V dc to the PIC's $\overline{\text{MCLR}}$ pin. You could connect 5 V dc directly to this pin, but it's better to apply it through a 10 kΩ current-limiting resis-

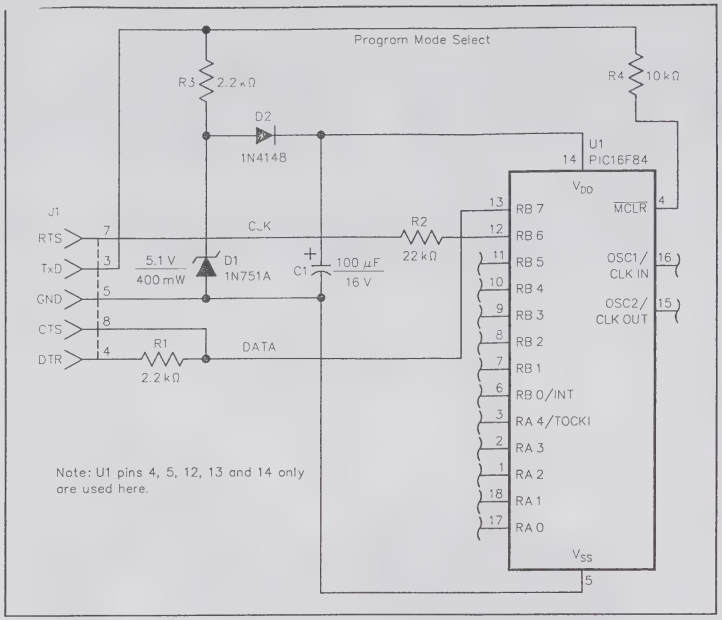


Figure 2—The simple PIC programmer schematic. Unless otherwise specified, resistors are ¼ W, 5% tolerance carbon-composition or film units. Equivalent parts can be substituted. RS part numbers in parentheses are Radio Shack; DK numbers are Digi-Key.

C1—100 µF, 16 V electrolytic (DK P1119)
D1—5.1 V, 400 mW Zener diode (DK 1N5231BDICT; RS 276-565)
D2—1N4148 or 1N914 (DK 1N4148DICT; RS 276-1122)
J1—D89
U1—PIC16F84 microcontroller (see PIC Resources sidebar)

Table 1
An Assembly-Language Program to Create a Code Practice Oscillator (CPO.ASM)

```
        list        p=16F84
        __config    0x3FF3
portb   equ         0x06
        org         0x000
        movlw       0x00
        option
        movlw       0xFD
        tris        portb
start   btfsc       portb,4
        goto        start
        bsf         portb,1
        bcf         portb,1
        goto        start
        end
```

tor; then you can restart the PIC's program by simply shorting U2 pins 4 and 5. If you try this without the resistor, you'll create a short circuit across your power supply. You will find that 10 kΩ resistors are extremely common in PIC circuits. They are used to limit current flow. When you apply 5 V to a 10 kΩ resistor, a current of 0.5 mA flows (E/R = 5/10000 = 0.0005). This is sufficient current for the chip to detect the signal, offers minimal power supply drain and is nowhere near a current level that could damage the PIC. If you are going to do extensive work with PICs, buy 10 kΩ resistors in bulk. Because the PIC is a CMOS device, it is a good idea to tie each *unused* input pin to +5 V through a 10 kΩ resistor in your final design. For experimentation purposes, this isn't necessary.

I'll get to the programming of the PIC later on. (First, we need a programmer for the PIC.) When it comes time to build the circuit of Figure 1, I suggest you do so using a solderless breadboard (Radio Shack and other suppliers have them.) This will provide you with a test bed for experiments and trial designs for your circuits. Using this basic test bed, you will be able to build a circuit with any combination of inputs and outputs you would like. Once you have finalized your design, you can transfer it to a protoboard or a PC board.

Programming Hardware

Before you can use the PIC in the circuit of Figure 1, you must program it for the task you want it to accomplish. In some instances, the program might be written for you. Authors of many *QST* projects that use PICs (and other micros) make the source code available. So, even if you have no interest in writing your own PIC programs, it may be useful to have a PIC programmer to burn chips using code written by others.

There are a number of commercially made PIC programmers available, some of which cost more than $150. ITU Technologies makes a very nice programmer kit that will program a wide range of PICs. It's available for $39 ($59 assembled and tested) and comes complete with programming software to get you up and running quickly.[13]

For those who like to roll their own, Figure 2 shows an incredibly simple PIC 16C84/16F84 programming circuit. It is based on a design by Ludwig Catta, with a few changes made to ensure that all parts can be purchased at Radio Shack. The device is powered directly from your computer's serial port, so no other power supply is required. If all the parts are purchased new, this programmer still costs less than $5 to build. You will need a socket to hold the PIC while it is being programmed. You can use a standard 18 pin DIP socket if you're simply burning a single chip based on someone else's program code. However, if you're planning to use the programmer to do your own development work (requiring frequent insertion and extraction of the PIC), buy a zero-insertion-force (ZIF) socket to minimize the wear and tear on the chips.

Software Needed to Use the Programmer

The code (program) that is eventually loaded into the PIC consists of a series of hexadecimal numbers collectively called *machine language*. Generally speaking, it's extremely difficult to actually write programs in machine language, so virtually all designers use an intermediate step: They write the software in *assembler* or a *high-level language* (such as *BASIC* or *C*). They then use an *assembler* or *compiler* to convert the code to the machine language that the PIC can understand. You can usually determine which type of code is in a file by the file name's extension:

- HEX—machine language files that can be loaded directly into the PIC
- ASM—files written in assembler. They must be converted into HEX before they can be loaded into the PIC.
- C, H—files written in *C*. They must be compiled before they can be used. Depending on the compiler used, they might be converted into .ASM files, or directly to HEX.
- BAS—files written in *BASIC*. They must be compiled before they can be used.

For *BASIC* and *C* files, you cannot use just any compiler to convert them into HEX files. You must use a compiler *specifically designed* to generate instructions that can be read by the PIC.

The most popular tool used to convert

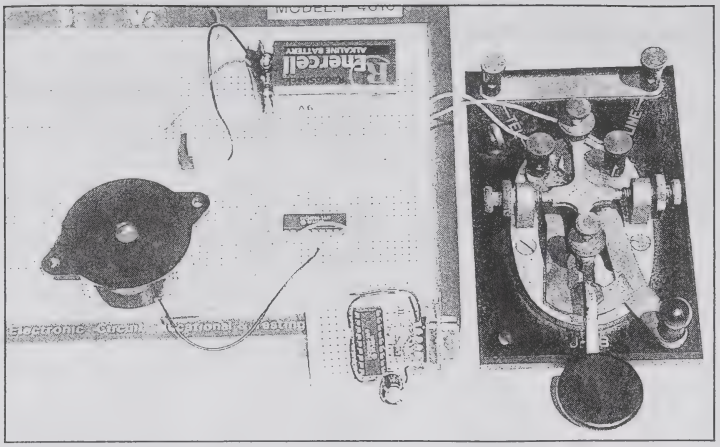
Table 2
An Assembly Language Program for a PIC Morse Code Generator

```
                list        p=16f84
                 radix      hex
                 __config   0x3FF1
dahlen          equ         d'99'  ;  <==controls code speed..
                                       ;increase
ditlen          equ         d'33'  ;  <==(though not over 255) for
                org         0x000       ;slower code, decrease for
                MOVLW       00          ;faster code. Make sure the
                MOVWF       0A          ;top number is three times
                                        ;the bottom.
                GOTO        start
                NOP
                NOP
time            MOVF        10,W
                BTFSC       03,2
                GOTO        endtime
uptop           MOVLW       01
                MOVWF       0D
upagn           CLRF        0C
doagn           DECFSZ      0C,F
                GOTO        doagn
                DECFSZ      0D,F
                GOTO        upagn
                MOVLW       4A
                MOVWF       0C
upone           DECFSZ      0C,F
                GOTO        upone
                DECFSZ      10,F
                GOTO        uptop
endtime  RETLW  00

dah             MOVLW       01          ;subroutine to do a dah
                MOVWF       0E
agn3dah MOVLW               dahlen
                SUBWF       0E,W
                BTFSC       03,0
                GOTO        enddah
                BSF         06,1    ;turn on pin B1
                MOVLW       01
                MOVWF       10
                CALL        time        ;wait 1 millisecond
                BCF         06,1        ;turn off pin B1
                MOVLW       01
                MOVWF       10
                CALL        time        ;wait 1 millisecond
                MOVF        0E,W
                INCF        0E,F
                GOTO        agn3dah  ;loop up to do it again.
enddah          MOVLW       3C          ;add a small delay
                MOVWF       10
                CALL        time
                RETLW       00

dit             MOVLW       01          ;subroutine to do a dit
                MOVWF       0E
agn3dit         MOVLW       ditlen
                SUBWF       0E,W
                BTFSC       03,0
                GOTO        enddit
                BSF         06,1    ;turn on pin B1
                MOVLW       01
                MOVWF       10
                CALL        time    ; wait 1 millisecond
                BCF         06,1    ;turn off pin B1
                MOVLW       01
                MOVWF       10
              CALL          time        ;wait 1 millisecond
                MOVF        0E,W
                INCF        0E,F
                GOTO        agn3dit     ;loop up and do it again.
enddit          MOVLW       3C          ;add a small delay
                MOVWF       10
                CALL        time
                RETLW       00

lspace                                  0xB4    ;subroutine to make a
                MOVLW                           ;letter space
                MOVWF       10
                CALL        time
                RETLW       00

start           CLRF        04          ;MAIN PROGRAM
                                        ;STARTS HERE
                MOVLW       0xFD
                TRIS        6

top             CALL        dit         ;modify this code to
                                        ;xmit the
                CALL        dah         ;CW you want to send
                CALL        dah
                CALL        lspace      ;lspace is a pause for
                                        ;the space between
                CALL        dit         ;letters. Include it after
                                        ;each letter.
                CALL        dit         ;As it appears here, the
                                        ;IDer will
                CALL        dah         ;transmit W2FS
                CALL        dah
                CALL        dah
                CALL        lspace
                CALL        dit
                CALL        dit
                CALL        dah
                CALL        dit
                CALL        lspace
                CALL        dit
                CALL        dit
                CALL        dit

                MOVLW       01
                MOVWF       0E
                                        ;this code programs the
                                        ;delay between IDs.
loop            MOVLW       d'150'  ;<= 4 x this number =
                                        ;number of seconds
                SUBWF       0E,W        ; between IDs when
                                        ;using a 4 MHz resona-
                                        ;tor.
                BTFSC       03,0
                GOTO        bottom
                MOVLW       10
                MOVWF       0F
again           MOVLW       0xFA
                MOVWF       10
                CALL        time
                DECFSZ      0F,F
                GOTO        again
                MOVF        0E,W
                INCF        0E,F
                GOTO        loop
bottom          GOTO        top
                END
```

Creating a PIC-Based IDer

Table 2 contains the assembly-language code that makes a 16F84 PIC generate Morse code. The code is designed to work with the circuit of Figure 1, using a ceramic resonator or a 3.5795-MHz colorburst crystal. Audio output is obtained at pin RB1. The program is designed to be easy to modify to insert your choice of call sign, Morse code speed and transmit interval. To change the Morse code speed, alter the lines labeled "dahlen" and "ditlen." To change the ID interval, change the number between the apostrophes in the line labeled "loop." Change the lines that start with the label "top" to change the call sign (or other text) that is to be transmitted by the chip. Follow the pattern shown in the program and make sure you put an "lspace" after the end of each letter. If you plan to transmit text other than just a call sign, you can do this by adding extra "lspace" instructions to create longer delays between the words. When altering this code, be sure to keep the label "top" on the same line as the first code element to be sent.

If you type in the program yourself, you may leave out the comments on each line by dropping the semicolon and the text that follows it. Blank lines can either be left in or deleted as you prefer. Alternatively, you can download this file from the ARRL's FTP site and edit it. The program may look long, but it only uses about 10% of the program capacity of a 16F84.



CPO on a breadboard—keyed by one of the famous J-38s. A perfboard version of the Ludwig Catta (Ludi) PIC programmer is in the foreground.

assembly language files to HEX files is available from Microchip itself. Called *MPASM*, it is easy to use—best of all, it is *free*! You can obtain *MPASM* from Microchip's Web site (see note 14). You can write the assembly language instructions using any editor that handles plain ASCII text (such as *Windows' Notepad* or DOS's *Edit*), then use *MPASM* to compile the code into a HEX file.

You also need software to load the HEX file into the PIC. The best program I've found for doing this is called *PIX* (see note 15). It supposedly is a DOS program, but I've had trouble getting it to run when my computer is booted in MS-DOS! However, *PIX* runs fine when I run it in a DOS box within *Windows* (that's a switch!). To get *PIX* to work with the programmer of Figure 2, first edit the *PIX.CFG* file that comes with the program. Find the two lines that say:

Port=LPT1

and

Programmer = Shaer

Place a semicolon at the start of each line. Then find the line that says:

:Programmer=Ludi

and remove the semicolon. Next, add a line that says:

Port=COM*x*

where *x* is the number of the serial port to which your programmer is connected.

It is important to use a *short* serial cable to connect your programmer to your computer. In building the programmer in Figure 2, I simply glued a female DB-9 connector onto the programmer itself and then plugged the programmer directly into the serial port on my computer. In any case, do not use a cable longer than about a foot.

When you run the program with the simple PIC programmer hooked up to your serial port, you will probably see a dialog box that says, **MODEM DETECTED** OR **NO/BAD HARDWARE. NOT TRUE CONTINUE.** Highlight the answer **YES** and press **ENTER**. The program will then start. This program allows you to read and write code to PICs and erase PICs. It will even disassemble the HEX code to show you the assembly language instructions. Pressing function key **F3** allows you to load your HEX file into the *PIX* program. When you press the **F9** key, the program loads into the PIC.

In summary, you use an ASCII text editor to write your assembly language instructions and save the file with an ASM extension. Then, use *MPASM* to assemble the ASM file into a HEX file. Finally, use *PIX* to load the HEX file into your PIC. To test your program, simply plug the programmed PIC into the completed target circuit and apply power. It should automa-

PIC Resources

Books

Benson, David, *Easy PIC'n: A Beginner's Guide to using PIC 16/17 Microcontrollers* (Kelseyville, California: Square 1 Electronics, 1996). It's a good introduction to PIC chips and assembly programming.

Benson, David, *PIC'n Up the Pace: PIC 16/17 Microcontroller Applications Guide* (Kelseyville, California: Square 1 Electronics, 1997). Essential reading if you are serious about learning assembler. If you plan to use a *C* or *BASIC* compiler, you need not have this book.

Predko, Myke, *Programming and Customizing the PIC Microcontroller* (New York, McGraw-Hill, 1998). Somewhat more advanced, this book contains a good discussion of the PIC architecture and lots of projects.

Peatman, John B., *Design With PIC Microcontrollers* (New York: Prentice Hall, 1997). This is a well written college-level text on PIC microcontrollers.

PIC Programmers

PIC-1A—Parallel-port programmer for all 16x6/16x7/16x8 chips. Available from ITU Technologies, 3704 Cheviot Avenue, Suite 3, Cincinnati, OH 45211, tel 888-448-8832, 513-661-7523, fax 513-661-7534; e-mail **sales@itu-tech.com**; Web site **http://www.itu-tech.com**. Kit: $39; wired and tested, $59. Add $9 for a ZIF socket. I have found this firm's customer service to be excellent.

For about $20, you can build more complex programmers that are similar to the ITU unit. See, for example David Tait's Web pages:

http://www.man.ac.uk/~mbhstdj/files/index.html
http://www.man.ac.uk/~mbhstdj/piclinks.html

Compilers

PCM, a *C* compiler available from Custom Computer Services, Inc, PO Box 2452, Brookfield, WI 53008; tel 414-781-2794 ext 35; **http://www.ccsinfo.com/picc.html**; price, $99.

Hi-Tech's PIC *C* compiler. Although this compiler is too expensive for most of us ($850), a working demo is available for free that will compile small projects. HI-Tech Software, LLC, Suite 105, 7830 Ellis Rd, Melbourne, FL 39204; tel 800-735-5717; **http://www.htsoft.com/**.

PicBasic Compiler, a *BASIC* compiler available from microEngineering Labs, Inc, Box 7532, Colorado Springs, Colorado 80933; tel 719-520-5323; **http://www.melabs.com/mel/**. Price: $99.95.

PIC Chips and Other Parts

Digi-Key Corporation, 701 Brooks Ave S, Thief River Falls, MN 5601-0677; tel 800-344-4539; **http://www.digikey.com**. Digi-Key also stocks ceramic resonators and ZIF sockets.

ITU Technologies, see information above.

JDR Microdevices, 1850 South 10th St, San Jose, CA 95112-4108; tel 800-538-5000; **http://www.jdr.com**.

[Author's note: This is not an exhaustive list of resources. There is a wide range of resources available for PICs including complete development environments that sell for over $2000. I have focused here only on those resources that are within the budget of a typical amateur.—*John Hansen, W2FS*]

Here's the homemade PIC programmer. FAR Circuits makes PC boards for this project and the others in this article.

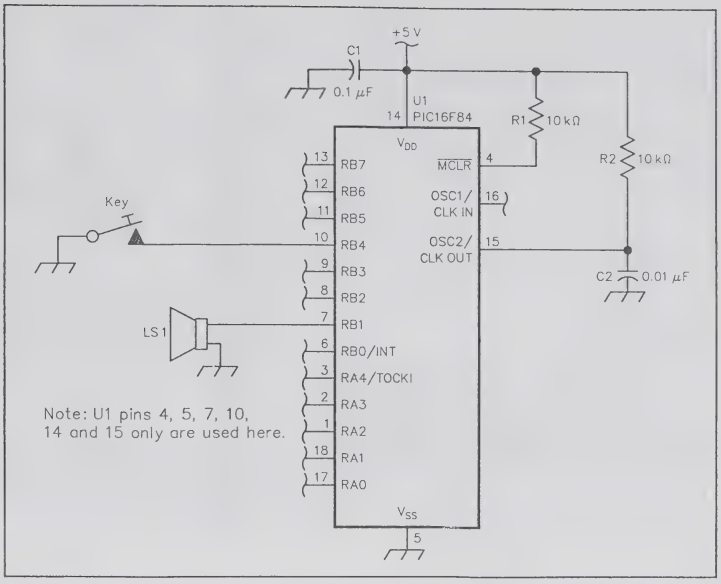


Figure 3—Code-practice oscillator schematic using a PIC microcontroller. Unless otherwise specified, resistors are 1/4 W, 5% tolerance carbon-composition or film units. Equivalent parts can be substituted.

LS1—Same as used in the circuit of Figure 1.

U1—PIC16F84 microcontroller (see PIC Resources sidebar)

tically begin executing its program.

Building Your First Program

Now that we can program the PIC, we need a program to make it do something. Learning to write assembly-language programs for the PIC is no trivial matter. My intention here is not to provide a tutorial in assembler, but to give you a sense of what it is like.[16]

The circuit of Figure 3 (quite similar to that of Figure 1) can be used to construct a simple code-practice oscillator (CPO). Sure, there are simpler ways to build a CPO than using a computer-on-a-chip, but this application does provide a good introduction to programming PICs. Furthermore, you can build this project using only two resistors, two capacitors, the 16F84 PIC and a piezo speaker, making this just about as inexpensive a way to build a CPO as the more traditional methods. The CPO circuit of Figure 3 uses an RC timing circuit for the clock. The key is connected between pin RB4 and ground. The speed at which the processor runs can be altered by changing the value of C2. Any capacitance value between 220 pF to 0.01 µF will work, with lower capacitance values resulting in a higher-pitched oscillator.

Using your text editor, enter the code given in Table 1 and name the saved file CPO.ASM. Enter the code in three columns as shown. You don't need to have the exact spacing between columns shown above, but you do need to have your code in three columns. Note the two underscores preceding the word "config." In Table 1, they appear as one long underline.

The first line of code tells the assembler which PIC is being used. In the second line, the internal configuration of the clock and the timers is set up. Here we specify that we are using a RC circuit for the clock. If we wanted to use a colorburst crystal or a ceramic resonator, we would have specified 0x3FF1.

The third line makes the code easier to read. The internal location for the Port B I/O lines (RB0 through RB7) is 0x06. This line of code specifies that we will call this location "portb" instead. The "org" instruction in the next line says where the program should start; in this case, is at the first instruction entered in the chip (at location 0).

The next two lines enable internal pull-up resistors on all of the Port B pins that are used as inputs. This means that 5 V is applied to each of these pins through a current-limiting resistor. This happens inside the chip and requires no external components. As a result, each of the input lines sees a 5 V signal, unless you short the pin to ground. When the line is grounded, the pin no longer sees 5 V. This is the mechanism used for keying the CPO. When you key the CPO, it momentarily connects pin RB4 to ground. The program detects that this pin is low, and generates a tone as long as the pin is grounded.

The next two lines determine which of the PIC's pins will be inputs, and which will be outputs.[17] The PIC assumes the pins are inputs, unless it is specifically told to make them outputs. In this case, the value FD (11111101 in binary) changes RB1 to an output. The next line is labeled "start" so that the program can loop back up to this line when it needs to. Once it gets to this point, all the program does is repeatedly execute the remaining lines of the program (except the end statement). This forms what is called an *infinite loop* because the program just continues doing this until you shut off the power to the microprocessor. In most programming environments infinite loops are avoided at all costs—they are often the things that cause computers to "hang" when they occur unintentionally. With PICs, however, infinite loops are very common. They are used whenever you want the PIC to continue running the same program over and over again until the power is shut off.

The line labeled "start" says that if pin four on Port B (RB4) is low (grounded), skip the next line of code. That next line just sends the program back to "start." Thus, if pin RB4 is high (+ 5 V), the program continues to alternate between these two lines of code until pin RB4 is grounded. When pin RB4 is grounded, the instruction to go back to "start" is skipped. The next two lines of code take pin RB1 high (bsf) and low (bcf) again. Then the program goes back up to "start." The effect of this is that if pin RB4 is grounded, the program causes pin RB1 to alternate between 5 V and ground at the same rate that the clock of the microprocessor is running. This produces a rectangular wave (not quite square) at about 700 Hz. Because we are using an RC circuit to clock the chip, it runs at a relatively low frequency. If we had used a 4 MHz crystal instead (which runs the microprocessor clock at 1 MHz), it would have been necessary to insert additional delay instructions to slow down the rate at which pin RB1 alternates between high and low. A CPO that runs at 1 MHz is not very useful! You won't hear it!

That's all there is to it. After you have saved the CPO.ASM file, use *MPASM* to

compile it into a HEX file using the command: **MPASM CPO**

If any errors are reported during the compiling process, it means you have mistyped something. By viewing the *CPO.ERR* file, you can find out which lines contain the errors. After the program assembles without error, use *PIX* to load the HEX file into your 16F84 PIC.

When you get this project running, try changing the code to lower the tone. Or, try having the PIC light an LED (as in Figure 1) *and* generate a tone. Each time you change the program, you need to rerun *MPASM* to reassemble the code and reload it into the chip with *PIX*. By experimenting with variations of this basic circuit, you can better understand how assembly-language instructions work.

Isn't There an Easier Way?

Of course, there is an easier way to do this. You can use a high-level language such as *C* or *BASIC* to write your program. For that, however, you need a compiler that converts your *BASIC* or *C* code into assembler or machine language. The cheapest of these compilers costs about $100. However, there are indications that some shareware compilers are beginning to come into the market. I've had very good luck with the Custom Computer Services, Inc compiler called *PCM*. It costs just under $100, and has built-in routines to make serial communication particularly easy. If you are going to do a lot of work with PICs, spending the money on a compiler may be worth considering.

Time to PIC Up Your Toys

What I enjoy most about PICs is that they allow me to combine my interests in hardware and software development. The circuits themselves are much easier to design because most of the heavy lifting is done by the code inside the PIC, or by specialized chips that can easily be mated to the PIC. The software isn't all that hard to write either, especially if you use a higher-level language and a compiler. This makes it possible for those of us with little or no formal background in either hardware or software engineering to do some amazing things with these inexpensive chips. Give it a try! You will find experimenting with these new toys to be an enormous amount of fun!

Notes

[1]Jeff Otterson, N1KDO, Peter Gailunas, KA1OKQ, Richard Cox, N1LTL, "Build a $60 Talking Repeater Controller," *QST*, Feb 1997, pp 37-40.

[2]Bryan H. Suits, WB8WKN, "DTMF/LT Decoding Made Easy," *QST*, Apr 1997, pp 34-36.

[3]Jay Craswell, WB0VNE (now W0VNE), "The PortaPeater," *QST*, Apr 1997, 37-39.

[4]Neil Heckt, "A PIC-Based Digital Frequency Display," *QST*, May 1997, pp 36-38.

[5]Lee Richey, WA3FY, "The CycleMaster," *QST*, Sep 1997, pp 37-42.

[6]Gary M. Diana, Sr, N2JGU, and Bradley S. Mitchell, WB8YGG, "TiCK-2—The Tiny CMOS Keyer 2," *QST*, Oct 1997, pp 42-45.

[7]Ron Alspaugh, W6NKS, "A Computer Keyboard CW Encoder," *QST*, Dec 1997, pp 32-35.

[8]Bob Anding, AA5OY, "A PIC of an IDer," *QST*, Jan 1998, pp 36-38.

[9]Steven C. Hageman, "Build Your Own Network Analyzer—*Part 1*," *QST*, Jan 1998, pp 39-45; *Part 2*, Feb 1998, pp 35-39.

[10]John Hansen, W2FS, "An Inexpensive, Remote-Base Station Controller Using the Basic Stamp," *QST*, May 1998, pp 33-37.

[11]Microchip Technology, 2355 West Chandler Blvd, Chandler, AZ 85224-6199; tel 602-786-7200, fax 602-899-9210; **http://www.microchip.com**.

[12]Because these chips are so similar, any project you find that uses a 16C84 (such as the Talking Repeater Controller; see Note 1) can be built with a 16F84.

[13]ITU Technologies, 3704 Cheviot Ave, Suite 3, Cincinnati, OH 45211; tel 888-448-8832, 513-661-7523, fax 513-661-7534; e-mail **sales@itutech.com**; Web site **http://www.itutech.com**.

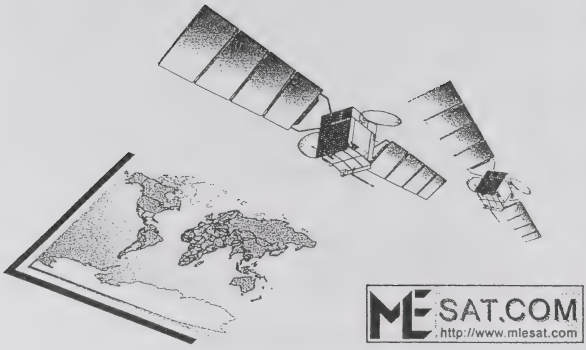
[14]MicroChip Technology, **http://www.microchip.com**. At that site, you will also find a free program called *MPLAB*. It is a complete development environment including an editor, a simulator and an assembler.

[15]Available at **http://home5.swipnet.se/~w-53783/**.

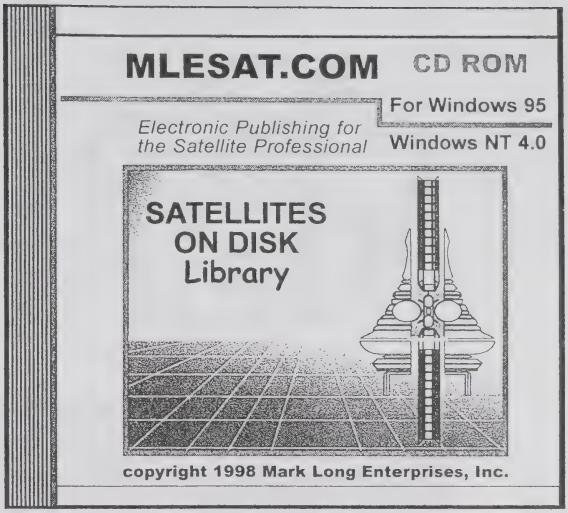
[16]For a good beginner's tutorial in assembler for the PIC, see David Benson, *Easy PIC'n: A Beginner's Guide to using PIC 16/17 Microcontrollers*, (Kelseyville, California: Square 1 Electronics, 1996).

[17]Microchip considers the *option* and *tris* instructions outdated. However, they will work in most applications. Because the alternative methods for configuring the I/O pins and pull-up resistors are somewhat more complicated, in the interest of simplicity, the first approach is selected for this project.

Formerly licensed as WA0PTV, John Hansen, W2FS, has been an Amateur Radio operator since 1966. His interests include satellites, digital communication and "homebrewing." For a number of years, John was the editor of The AMSAT Journal, *and is the 1993 recipient of the ARRL Atlantic Division Technical Achievement Award for his work on digital-satellite-gateway software. John maintains a career as a Professor of Economics at the State University of New York to provide enough income to purchase radio equipment and PICy things. You can contact John at 49 Maple Ave, Fredonia, NY 14063; e-mail* **hansen@fredonia.edu**.



Full colour footprint maps, frequency plans, spacecraft illustrations, transponder guides and contact information with supporting charts, graphs and photos for all the present and future telecom satellites.


MLESAT.COM CD ROM
For Windows 95
Electronic Publishing for the Satellite Professional
Windows NT 4.0
SATELLITES ON DISK Library
copyright 1998 Mark Long Enterprises, Inc.

It used to be that anyone needing access to technical information on telecommunication satellites had to wade through high-priced reference books containing hundreds of pages of text and technical drawings. Even then, the information in these vast tomes was out of date by the time the reader actually received it. The **SATELLITES ON DISK LIBRARY** allows you to access comprehensive, up-to-date technical information on geostationary communications satellites directly from your personal computer. What's more, you can quickly locate the graphic or text-based information you need, import it into your favourite word processing and page layout programs, and then add it to your own in-house reports, marketing studies, advertisements, customer proposals, and training materials. Best of all, you don't have to wait for anyone to print the next annual installment of their pre-historic paper publications because the **SATELLITES ON DISK LIBRARY** includes a software upgrade option which will allow you to download the latest updates from our Internet corporate site on the Worldwide Web.

Covering Asia, Australia, Europe, The Middle East and the Pacific Rim
All present and future satellites from 30 West to 183 East Longitude

The EURO-Asia/Pacific Edition SATELLITES ON DISK LIBRARY:

- 375 full-colour satellite footprint maps
- dozens of satellite line drawings
- 500+ pages of technical descriptions
- regional satellite summary charts
- transponder plans & activity charts
- other essential reference materials
- Runs on IBM PC compatible computers running Windows 95. Use Internet Explorer or Netscape Navigator to to fully explore all HTML links.
- Easy to understand Installation manual.
- Optional Satellites-On-Line update service at http://www.mlesat.com

EURO-Asia/Pacific SATELLITES ON DISK LIBRARY
US$ 129.95 plus $15 shipping & handling. All checks must be in US dollars drawn on a US bank and made payable to MLE INC. Fax credit card orders or mail form together with payment to:

MLE INC. P. O. Box 159, Winter Beach, Florida 32971 USA
Fax (1): 734-433-0935 **E-mail:** mlesat@mlesat.com

Name: _____
Company: _____
Address: _____
_____
City: _____ **State:** _____
Zip: _____ **Country:** _____
Tel: _____ **Fax:** _____
MC/Visa: # _____
Exp. Date: _____ **Signature:** _____

HELP US
bring

DR DISH

to SPRSCS '99!

Your DONATION goes 100% to the travel costs for Dr Dish to New Zealand and you will be acknowledged (unless requesting anonymity) on television!

Christian Mass is the most respected technical mind in home satellite TV in Europe. His position of Senior Editor with Germany's Tele-Satellit Magazine and his monthly "*Dr Dish*" TV show seen across Europe, Africa and the Middle East by satellite is the legendary "meeting place" of technology and new developments in our exciting field. His intimate knowledge of satellite receivers, encryption systems and the status of every aspect of home satellite TV in Europe makes him the most sought after expert in his field.

We are bringing Christian Mass to SPRSCS '99 so that he can share with us, live in person and throughout the balance of 1999 on our SPACE Pacific Report television programme, his vast experience in this field. Your donation of money to the "Dr Dish Fund" will help SPACE cover the cost of his travel to SPRSCS '99 and you will be acknowledged on our TV programme as well as the Dr Dish European TV programme for your donation!

Your donation can be by cheque, cash (please register envelope) or credit card!

☐ **YES** - *I wish to donate to the Dr Dish Travel Fund!*
- ☐ Please charge my credit card as below
- ☐ My cheque in the amount of $_____ (to SPACE Pacific/fund) is enclosed
- ☐ I am enclosing $_____ in cash

My name _____
Company name (if applicable) _____
Mailing address _____
Town/city _____
☐ I request that my donation **NOT** be made public (no TV listing!)

Charge my donation to my credit card as follows:

☐ VISA ☐ Mastercard number __ __ __ __ - __ __ __ __ - __ __ __ __ - __ __ __ __
Name on card _____ expires ____/____

Note: Return to SPACE Pacific, PO Box 30, Mangonui, Far North, New Zealand or if by credit card, fax to 64-9-406-1083

ATTENDANCE REGISTRATION FORM

South Pacific Region Satellite & Cable Show '99

Instructions:

First - Decide whether you will attend a Mark Long / SPACE Certificate Course

Second - Decide which course you will attend - or both if you are ready to knock 'em dead in 99!

Third - Fill in your name and other information requested, and then select the box that describes your attendance level

Fourth - Complete sections relating to lodging, your transportation schedule

Fifth - Select payment method and return to us using address at bottom of next page

SPACE Pacific / Mark Long Certification Courses

SATELLITE DIGITAL INSTALLER COURSE

Comprehensive beginner course that starts you at ground zero and rapidly advances you through a complete understanding of how digital television information is transmitted and processed by the receiver (IRD). Loads of course study materials, handouts, reference materials. Tests at end, certificate when you pass exams. Two days (**March 23 and 24**), fee is US$295 (note: same price as correspondence course only you have Mark Long teaching you live!).

SATELLITE DIGITAL TECHNICIAN COURSE

You start from the end of the "Digital Installer Course" (or an equivalent knowledge base from first hand industry experience - generally accepted as two years or more of digital installation work) and advance to a level of being functional in an uplink or sophisticated (cable TV, broadcast station) downlink environment. Course study and reference materials, exams and a certificate when you successfully complete testing. Two days (**March 26 and 27**), fee is US$395 (note: same price as correspondence course only you have Mark Long teaching you live!).

SPRSCS '99 - With or Without Attending A Mark Long Course

Two "formal" days (March 25 - 26) but you are invited to stay over March 27 as well because we will be "shooting TV programmes" from March 24 through March 27th. Be a part of something that will still be showing up on television ten years into the future - or just be an observer who can say, "*I was there when they .....* ." Bring your own camcorder and shoot as much as you wish of the TV programme creation. Become acquainted with the unique people who make this industry move so rapidly and advance your own knowledge of this specialised field.

The Summary Calendar

March 23 and 24: Mark Long / SPACE Digital Installer Course (plan to arrive March 22nd and if staying only for this course, leave after 5PM March 24).

March 25 and 26: SPRSCS '99 TV Programme "Shoot" (actually starts March 24 and ends March 27). Set your own arrival / departure schedule.

March 26 and 27: Mark Long / SPACE Digital Technician Course (plan to arrive March 25th and if staying only for this course, leave after 3PM March 27).

YOUR REGISTRATION DETAILS

Your name _____

Company (as applicable) _____

Mailing address _____

Town/city _____ _____ _____

Telephone no _____ Fax no _____

email address _____

I WISH TO ATTEND -

☐ Mark Long / SPACE Digital Installer Course March 23/24 at US$295 (*)

☐ Mark Long / SPACE Digital Technician Course March 26 & 27 at US$395 (*)

☐ SPRSCS '99 March 25 & 26 at NZ$200

(* or equivalent in NZ$ or A$ as of the date of registration)

☐ COMBO DEAL One - Mark Long / SPACE Digital Installer plus SPRSCS '99 at US$295 + NZ$150

☐ COMBO DEAL Two - Mark Long / SPACE Digital Technician plus SPRSCS '99 March 24 & 25 at US$395 + NZ$150

☐ COMBO DEAL Three - Mark Long / SPACE Digital Installer + Digital Technician + SPRSCS '99 (March 25 only) at US$295 + US$395 total

TRANSPORT - LODGING

☐ Please arrange lodging (indicated below) for the nights of (circle applicable dates)

March 22 23 24 25 26 27

☐ Single room ☐ Room for 2 people ☐ Other _____

with (leave blank if does not apply to you) ☐ full cooking facility

☐ beach front accomodations ☐ Other _____

I will arrive at ...

☐ Kaitaia (NZ) Airport at _____(AM/PM) on March _____; please pick me up

☐ Kerikeri (NZ) Airport at _____ (AM/PM) on March _____; please pick me up

PAYMENT METHOD

Please charge to my ☐ VISA ☐ Mastercard credit card (will be charged in US$ for Msark Long Courses at applicable conversion rate; NZ$ for SPRSCS '99) as follows:

__ __ __ __ - __ __ __ __ - __ __ __ __ - __ __ __ __

in name of _____

expires ___/___

Cheque in the amount of $_____ (to SPACE Pacific) enclosed

PLEASE - only ONE individual per form (make your own extra copies as required). Return to SPACE Pacific, PO Box 30, Mangonui, Far North, New Zealand or if by credit card fax to 64-9-406-1083.

YOU WILL HEAR FROM US -

confirming your registration and with complete details on your lodging accommodations. If flying to Kaitaia or Kerikeri, we will provide ground transportation from each to the event as well as back to the airport upon your departure.